RnP ELEX DOCUMENTATION

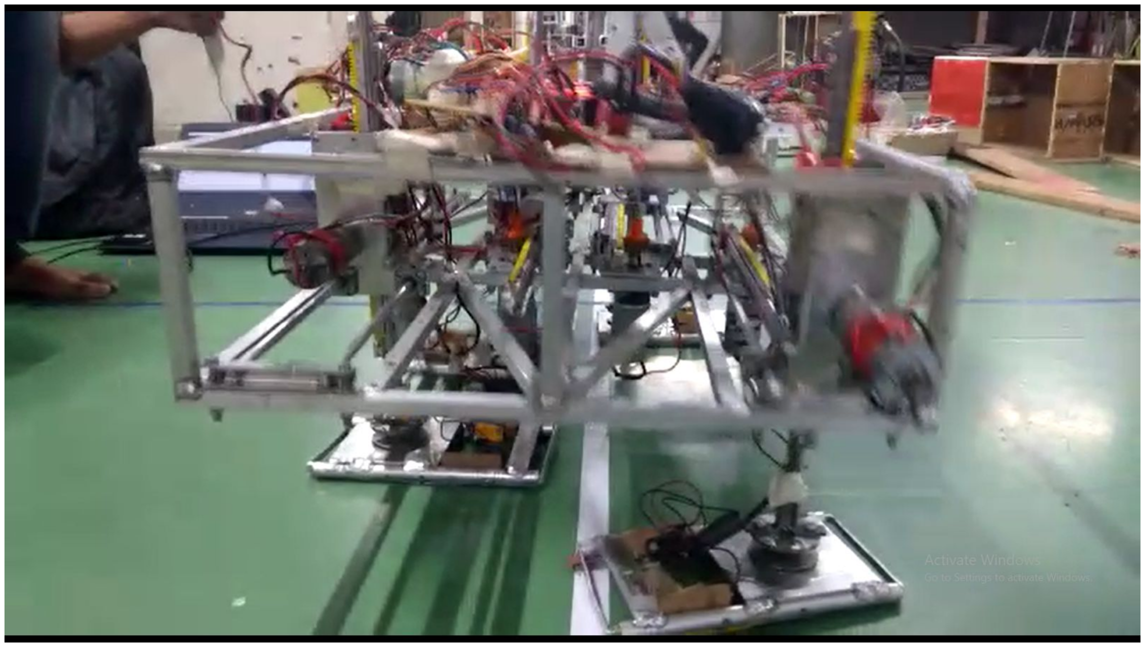
RnP : Messenger Robot of VIT Pune Team B. Based on linear motion via Rack and Pinion mechanism

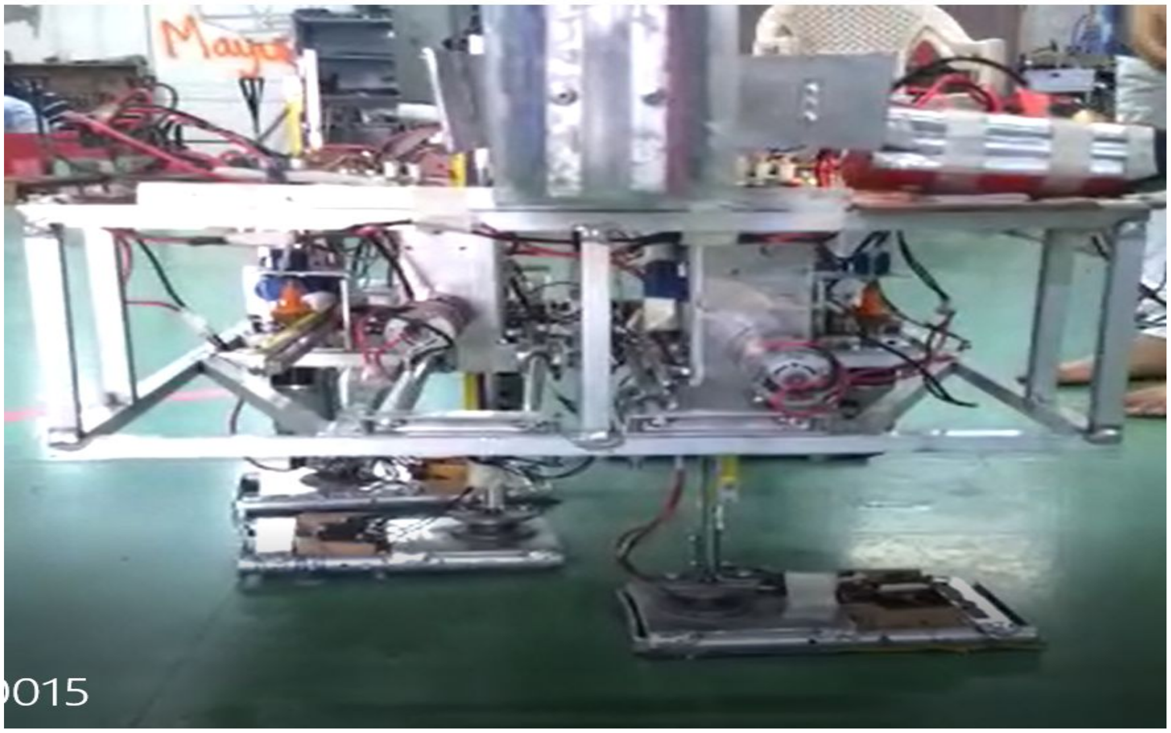
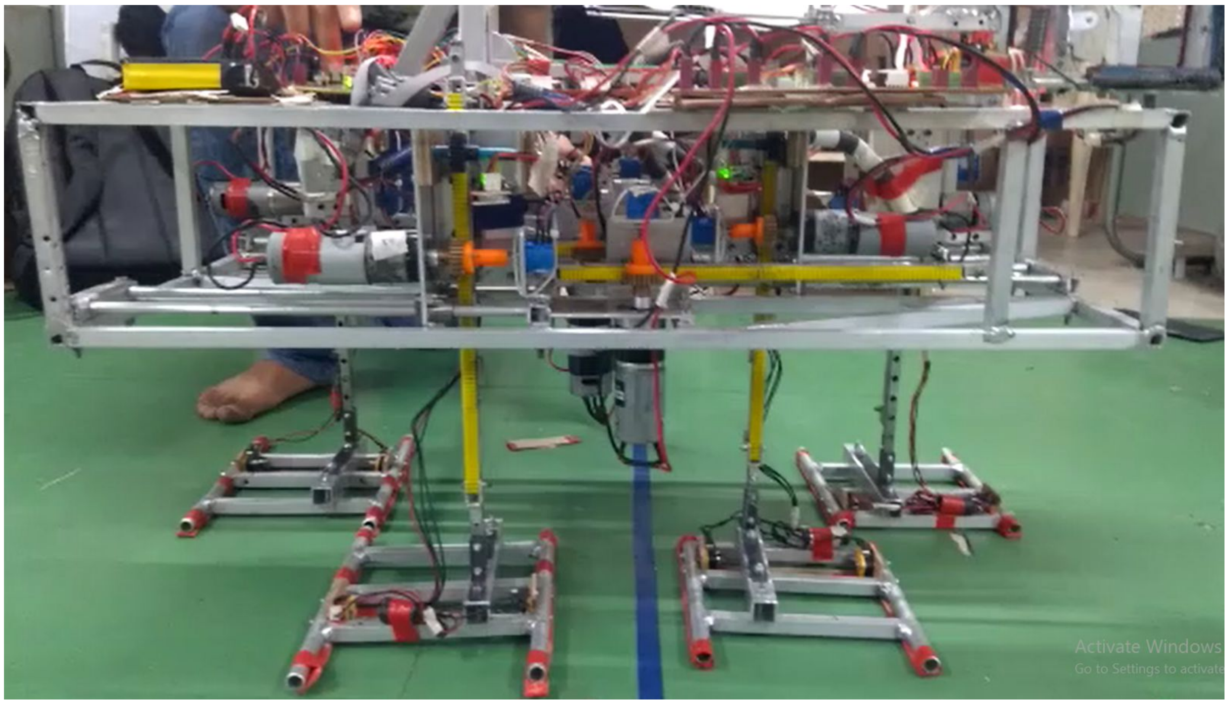
Major tasks which were serially done :

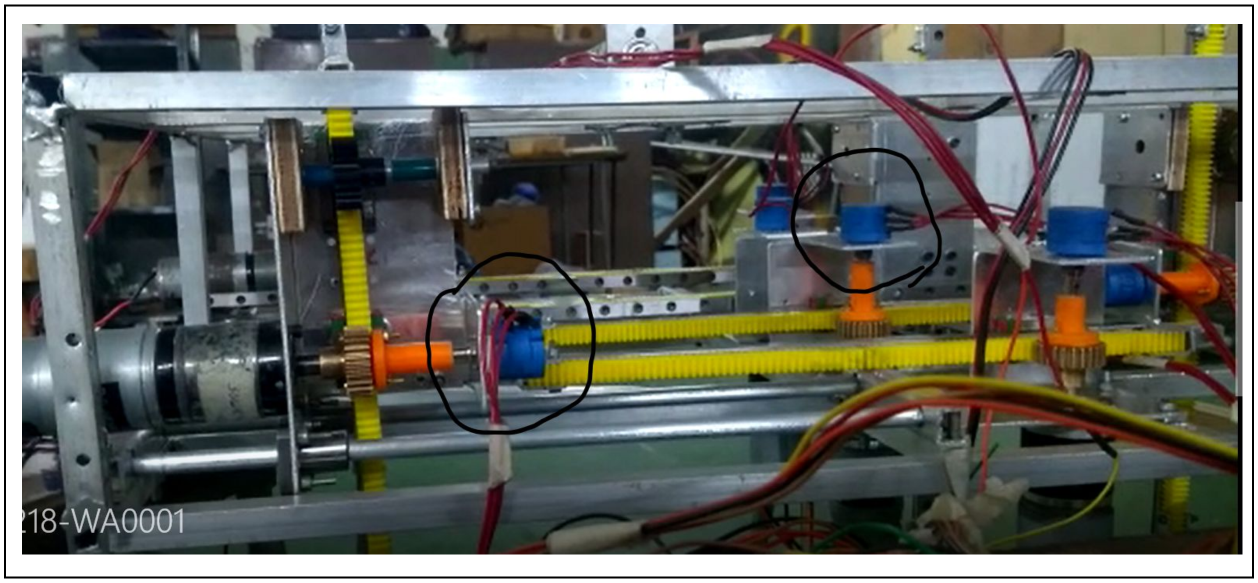
1) Position Control of Pinion on Rack using different Feedback.
2) Stable patterns for Walking and Turning. Code for the same.
3) Various Patterns for Turning (Blunt or Sharp) (Choice of Grip)
4) Developing Algorithm to cross Sand-dune (Height = 10cm and Width = 30cm) and Tussock (Two slack Nylon ropes).
5) Mountain Climbing with different Feet and Patterns (Angle of Inclination = 14.9°) and Uukhai.
6) Gerege detection Feedback

RnP Final Images:

Front View:



Back View: 
Side View: 


218-WA0001

Final Feedbacks used :
(Images are attached for reference, implementation and problems faced are mentioned)

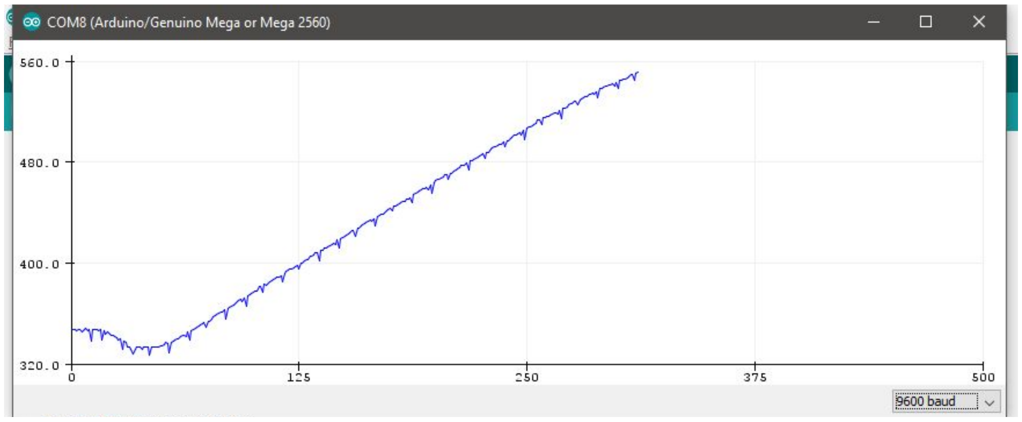
1) Multi-turn Potentiometer (10k Ohm) for each motor :

Implementation :

- For position control of Pinion on Rack : Total 8 Pots were used on bot with 2 on each leg (for vertical and horizontal motion Control)
- These were interfaced with 10bit ADC of Microcontroller so the value was observed between 0 to 1023 where 0 is observed at 0 turn and 1023 at 10th turn. While theses pots were mounted, they were mounted preferably at 5th turn.
- Connections (If shaft up):

Uppermost Terminal	Gnd (0V)
Middle Terminal	Vcc (+5V)
Lowermost Terminal	(Data Terminal) To analog pin of Controller

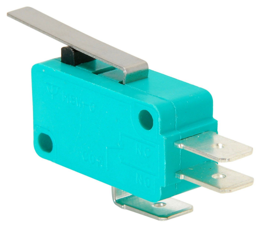
Problems Faced :
1.Slippage - Slippage was mainly observed between pot shaft and coupling (which were 3D printed). This used to affect the position control of motor and hence the Leg.This issue was solved later by changing the design and material of coupling.
2.Fluctuating Reading - Internal of Multi-turn Pot which were used,mainly consists of a moving ring and a 10 turn resistive wire on which the moving part used to come in contact and corresponding resistance is used as output.Due to transverse load on pot shaft,moving part used to touch upper and lower ring simultaneously and the output used to be fluctuating.Now due to ten turn pot and 10bit ADC,resolution was quite less.These two reasons affected the task to control small displacements.
3. Other issues like wear-out of potentiometer resistive ring was observed due to excessive Force which was being applied by the Motor.
Image attached below shows the fluctuating readings observed.



Measures to solve problems:

- Capacitors os 0.1μF were connected in parallel across the the data pin from where it was connected to and Ground in order to reduce the observed fluctuations.
  However the fluctuations were reduced but the reading frequency was slow downed i.e pots readings were still for some moments even though there was physical rotation on Pot's shaft.
- The FRC that connected the Pot board and Micro-controller was faulty. Changing this FRC saw reduced fluctuations.
- Since readings of pot were a major problems, we though there is drift in reference voltage while ADC. So, Aref of microcontroller was connected to 5V externally to solve this problem. No major change was observed even after doing so.
- Since pots used to move a little during motion. This caused then to given incorrect readings and more fluctuations were observed. So they were fitted with cable tie from top to eliminate the above problem.

2) Li**mit switches** :

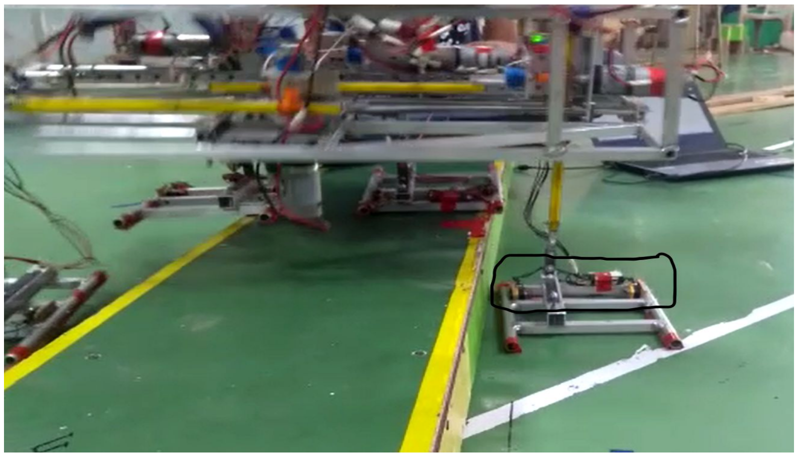


Implementation:

- Mounted on each foot and used near sand-dune to stop the vertical leg at intermediate position.
- Also mounted on gerege passing mechanism to detect if gerege is passed from MR1 to MR2. It was mounted on the mouse-trap mechanism used to hold the gerege. Then the trap is in loaded position,.this Limit switch would be pressed and then gerege is passed, this LS would get released and after certain delay (5seconds), the bot will start to walk.
- Two limit switches were mounted on each foot. One near the from and other one near the back. Also they were mounted close to
- The Common Terminal of limit switch was INPUT_PULLUP by the microcontroller and the normally open(NO) terminal was connected to ground. So when the limit switch is pressed, the common will get connected to NO of the Limit switch and Low reading will be observed. Else High reading will be observed by the controller

- Connections:

Common	To digital pin of microcontroller
Normally closed (NC)	Insulated (Not connected)
Normally Open (NO)	To Ground



<u>Problems Faced :</u>

- If the limit switches were mounted far from the point of contact of leg and slider then when leg was on surface then limit switches couldn't get pressed and hence feedback was not obtained.
- If the NC terminal was not properly insulated and gets in contact with the surface of leg then the potential difference will be spread on the leg and wrong readings were observed even though the switch is pressed.
- If the leg does not properly fall on the surface of sand-dune or only a little part falls on the surface then the limit switch may not get pressed

<u>Measures to solve the problem:</u>

- The limit switch was mounted properly on the leg, near to the point of contact of leg and the slider. This ensured that both the switches on each leg gets pressed
- A cardboard flap was used to connect the flaps of both the limit switch of each leg so that when one is triggered then because of cardboard flap, other one is also pressed.

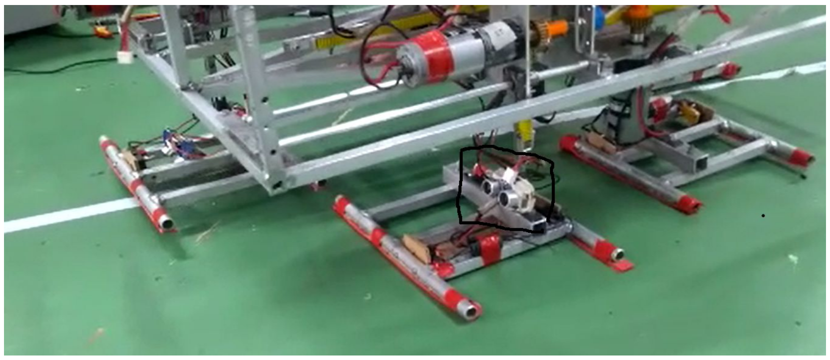
3) Ultrasonic sensor HC SR-04 :

<u>Implementation:</u>

- Used to detect sand-dune. Mounted on both the foot of front Legs. Position is shown as below. When the bot approaches Sand-dune, the distance is calculated between the Sand-dune and the foot whichever is further. This distance is then mapped and the displacement of rack is manipulated i.e step length is manipulated.

- Two such such sensors were mounted on each leg.
- Connections:

Vcc	+5V
Trig	To digital pin of microcontroller
Echo	
Gnd	Ground



<u>Problems Faced:</u>

- Noise in signal
- Improper mounting

<u>Measures to solve problem :</u>

- Proper Mounting

4) MPU6050:

<u>Implementation:</u>

For yaw and pitch. Yaw was used for turning till Sand-dune and tussock turn. Pitch was when the bot was going to climb the mountain.

As the bot approaches Sand-dune it has to take turn (left for red and right for blue) of 45°. The feedback of turn was provided by yaw readings of MPU.

For climbing the mountain (angle of elevation 15°), the bot has to climb till it reaches the top which is at 0 degree elevated from ground. So as after certain cycles pitch feedback was taken and if reading goes 0degree i.e bot has reached the top, it will execute further task.

Problems Faced:

- Initially, we were taking the float data serially from nano which was used to interface the MPU. But this communication was very slow.

Measures to solve the problem:

- Instead of taking the data serially, we decided to convert it into digital signal. 3 digital pins were taken. One for choosing yaw and pitch. Other two for direction status in binary coding.

5)Optical Proximity Sensor :

Implementation :

Used to start the robot without touching near mountain area which was required as per the rulebook. When the bot crosses tussock and reaches mountain base, it is stops and waits till MR1 scores 50points by throwing Shagai. As the desired score is achieved, the bot can start climbing the mountain and this signal is given by activating this proximity sensor. Simply waving in front of it will activate it and the bot will start to climb the mountain.

Alternatives tried for control of motor feedbacks :

IR SENSOR:

Implementation:

- The IR sensor was mounted in such a way that its transmitter and receiver used to face the side of the rack. Transmitter and receiver was covered with black tape leaving the front to be exposed to the side of the rack. This was done to avoid firing of sensor because of external lightings.
- White paper was stuck to the side of the rack to be detected by the IR Sensor.

Problems Faced:

- Since the sensor is an optical sensor, it required tuning with change in lighting conditions
- Also the mounting was difficult and required changes on the clamp
- Hence, this sensor was discarded

Limit switch on vertical rack:

Implementation:

- Limit Switch was mounted inside the clamp and a long bolt was mounted on rack such that it will fire the the Limit switch when leg goes up.
- This was done in order to avoid overshoot if pot gives incorrect readings

Problems faced:

- Difficult to mount
- The bolt would often get bent after certain cycles
- Two were supposed to be mounted on one leg but since mounting on single was difficult, this was not possible.

Hall Sensor:

Implementation:

- Hall sensors were used to stop the motion of the rack after the leg reaches a certain height or touches ground. The magnets were mounted on the rack accordingly. They were mounted on vertical racks only as secondary feedback in order to avoid overshoot if pot fails to provide correct readings. Later it was also used as primary feedback when pots were constantly providing incorrect readings.
- Hall sensors were mounted one on each leg inside the clamp close to the rack since magnet's and sensor's range is very small. They were interfaced at the **interrupt** pins of the microcontroller.
- Three magnets were mounted on the rack. One each at the extreme on the rack and one near the bottom such that if all the legs stop at that position, the bot's height will be low. This height was used at normal walking.
- The hall sensor used was **in-house Hall sensor module** as the module available in market triggers 2pulses when it detects the magnet.
- Since a certain face of the magnet triggers the sensor, so correct mounting the magnets was very important. Also it was essential to tune the sensor.
- Another method was implemented to take count (increment when going up and decrement when going down)
- Neodymium magnets were used as they have good range

Problems faced:

- Often the sensor would misfire. This would result in incomplete and improper motion of the vertical rack
- The counts used to get missed
- This is a digital feedback unlike potentiometer which is analog feedback. Therefore implementation of PID is not possible with Hall sensor. Thus, motor has to be operated at low speed.
- Later, they were not a reliable feedback.

Work done on prototype

1.Walking pattern

In diagonal pattern at a time 2 Legs(diagonal) were responsible to move the chassis forward.**2 types** of **Diagonal pattern** were implemented at various heights so that stability and speed is achieved :-

Type 1 : This Diagonal pattern was designed so that one cycle occurs in minimum number of states.It was used in Normal walking (at height of 18cm as mentioned earlier). Since the centre of gravity was Low due to lower Bot ht in Normal waking the weight shifting of Legs during this pattern didn't affected the bot's Stability.

Consider the Leg 1 be at front left corner and let the numbering be anticlockwise so that leg 4 is at front right corner. Let each leg have two horizontal positions – front and back, and two vertical positions – up and down.

Let the initial position be with all legs on the ground. And let leg 2 and 4 be at their front Position and Leg 1 and 3 at their back positions.

Hence the walking can be divided into **Six** main steps:

i. Leg 1, 3 - UP

ii. Leg 1, 3 - FRONT, Leg 2, 4 -BACK

iii.	Leg 1, 3	-	DOWN
iv.	Leg 2, 4	-	UP
v.	Leg 2, 4	-	FRONT, Leg 1, 3 -BACK
vi.	Leg 2, 4	-	DOWN

KEY FEATURES and observations :

- Since in step (ii) and step (v) all forward and backward motion of Legs are in parallel, total time for one cycle was optimized.
- As the Legs 1 and 4 are far away as compared to Legs 3 and 4 from **Length** axis of bot and also the momentum and weight shifting in step(ii) and (iv) is in forward direction,the bot used to topple in front left or right corner.At greater height this toppling affect was quite noticeable and used to cause unstability.

PROBLEMS FACED AND SOLUTIONS

Initially legs 2 and 3 were designated as front legs but due to sideways toppling of the bot it was decided to designate the legs 1 and 4 as front legs since they were towards the outer side of the bot.

2.Turning

Torsional Shoes were mounted on the foot of the bot. For turning the rack in a certian direction the racks in the opposite direction were given more displacement than the other two horizontal racks.

3. Sanddune

Sandune was crossed in diagonal pattern with LS used as feedbacks.

PROBLEMS FACED AND SOLUTION

It was observed that a single LS was not fired correctly if placed in the centre of the foot. This problem was solved by mounting 2 LS- one on the forward and one on the backward edge of the foot.

Motors used

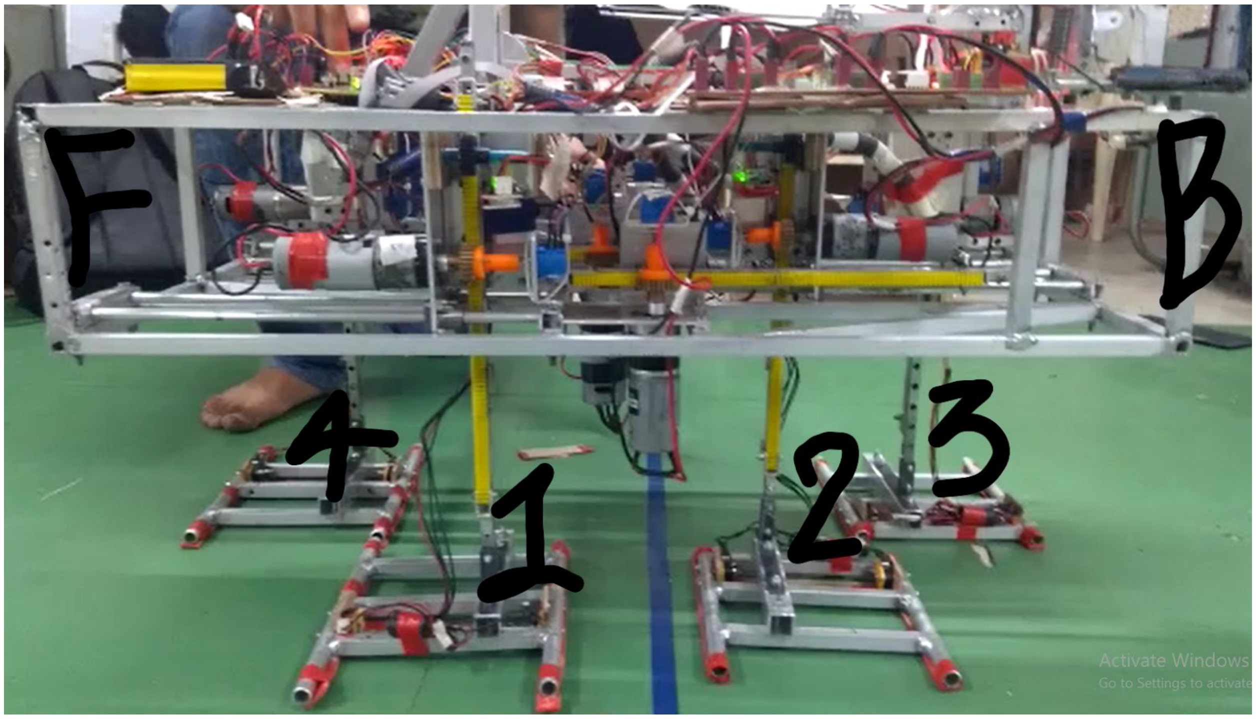
10rpm 40kgcm in vertical direction.
2 of 200 rpm and 2 of 300 rpm in horizontal direction.

Description of each task :

Terminologies :

front(F) : extreme end away from gerege mechanism.
Back(B) : extreme end near to the gerege mechanism.
Mid_front
mid_back
Down : All the Legs extended vertically such that the height of chassis from ground is 26cm.
Down-mid : All the Legs extended vertically such that the height of chassis from ground is 18cm.
Up
Upmid

Following figure indicates the Front(F) and Back(B) alongwith the Leg numbering terminologies.



1)Position Control of Motors :

In Total 9 DC motors were used :

2 for each Leg : Vertical and Horizontal Motion.

1 Motor for uukhai.

Main task was to control the position of any Leg independently(i.e:control of horizontal and vertical Motors).Once this task is achieved any leg can be brought to any position within the limits of the Rack.

Algorithm :

- PI(Proportional Integral) controller was used to control the position of Motors.
- To increase the speed total horizontal path was divided into three sections according to the PWM being applied,If the horizontal Movement in particular step is suppose **x** then :
    1. First 1/3rd(**0 to x/3**) part was assigned as acceleration in which the pwm was increased gradually by some factor and saturated to the constrain value .
    2. Middle part(**x/3 to 2x/3**) was given constant and Max constrain pwm.
    3. Last part(**2x/3 to x**) was decelearation and pwm was calculated using PI controller.
- For control of motors in above mentioned manner seperate functions were developed to control each Motor which to take the parameter as the desired position.Using the analog feedback of pot error was calculated between the current position and desired position.
- Potentiometer value for extreme positons are stored in unique variables for each motor.Since the Potentiometers gives absolute Feedback,even if Microcontroller is reset potentiometer will give approximately the same value and update of position variables need not to be taken care of whenever the MIcrocontroller reset happens.

2)<u>Stable patterns for straight walking and writing the code for same</u> :

1. **Three Legs on ground pattern :** In this pattern,three Legs are on Ground(stationary) and one of the Leg is lifted up and brought to the desired position according to the steps decided.This pattern was developed for stability of Bot and worked for smaller foot.
   Steps:

   1. Initial position : Initially all the legs are on ground and in front().
   2. Now force is applied in backward direction by all the 4 Legs ,by passing the back value of potentiometer to Motor control Function.Since the Legs are on Ground and Friction Force in Forward Direction causes the Chassis to move in Forward direction.
   3. All the legs are in back position now.It is required that now all the Legs to be in Front so that Initial Positoin to move the chassis forward is obtained.
   4. One by one all the four Legs are brought in front by lifting it up and moving it forward and again keeping it down again,at the same time the other three Legs are on ground and in rest.

2. **Diagonal pattern :** In diagonal pattern at a time 2 Legs(diagonal) were responsible to move the chassis forward.**2 types** of **Diagonal pattern** were implemented at various heights so that stability and speed is achieved :-

   Type 1 : This Diagonal pattern was designed so that one cycle occurs in minimum number of states.It was used in Normal walking (at height of 18cm as mentioned earlier). Since the centre of gravity was Low due to lower Bot ht in Normal waking the weight shifting of Legs during this pattern didn't affected the bot's Stability.

   Consider the Leg 1 be at front left corner and let the numbering be anticlockwise so that leg 4 is at front right corner. Let each leg have two horizontal positions – front and back, and two vertical positions – up and down.
   Let the initial position be with all legs on the ground. And let leg 2 and 4 be at their front Position and Leg 1 and 3 at their back positions.
   Hence the walking can be divided into **Six** main steps:

vii.	Leg 1, 3	-	UP
viii.	Leg 1, 3	-	FRONT, Leg 2, 4 -BACK
ix.	Leg 1, 3	-	DOWN
x.	Leg 2, 4	-	UP
xi.	Leg 2, 4	-	FRONT, Leg 1, 3 -BACK
xii.	Leg 2, 4	-	DOWN

   KEY FEATURES and observations :

   - Since in step (ii) and step (v) all forward and backward motion of Legs are in parallel,total time for one cylce was optimized.
   - As the Legs 1 and 4 are far away as compared to Legs 3 and 4 from **Length** axis of bot and also the momentum and weight shifting in step(ii) and (iv) is in forward direction,the bot used to topple in front left or right corner.At greater height this toppling affect was quite noticeable and used to cause unstability.

Type 2 : The Above(type 1) Diagonal pattern when used on greater height of chassis configuration,problem of weight shifting and unbalanced torque used to cause the unstability and toppling of bot.
Greater height was required to cross sanddune and tussock.So to use Diagonal pattern there as well , different states were designed which took care of stability as well as speed.

Let the initial position be with all legs on the ground. And let leg 2 and 4 be at their front Position and Leg 1 and 3 at their back positions.

Hence the walking can be divided into **Six** main steps:

i.	Leg 1, 3 - UP	
ii.	Leg 1, 3 - FRONT	
iii.	Leg 1, 3 - DOWN,	Leg 2,4 - BACK
iv.	Leg 2, 4 - UP	
v.	Leg 2, 4 - FRONT	
xiii.	Leg 2, 4 - DOWN,	Leg 1, 3 - BACK

KEY FEATURES and observations :

- Although the number of states was same as type 1 pattern,time required to complete those states were increased.
- Whenever the Legs were bought back the bot back,the bot used to tilt in front left or right corner according to configuration.This problem was solved by integrating **down movement of lifted diagonal legs with backward motion of other two diagonal legs**.

Code Structure :

This section will explain how the code was developed,maintained and algorithm or flow to sample sensor and actuating the actuators and state switching etc.

Code consists of :

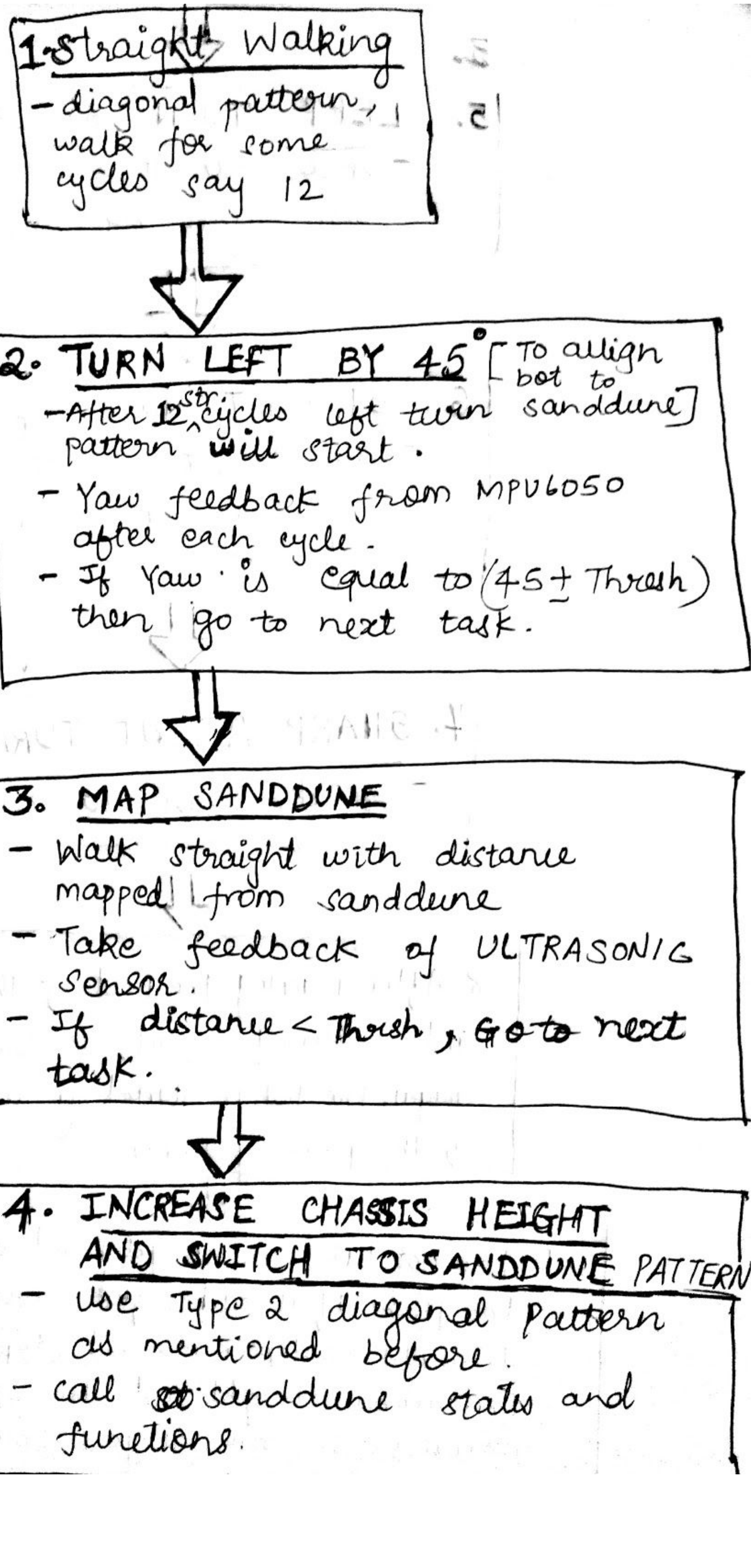
- Main .ino file where all the initialisation and state switching used to happen(setup and loop).
- Library files or header files(.h) were made for :
  1. **Each pattern** : Order in which states should switch according to the pattern.PI Control functions of that motors were called which were required to attain particular position for that state.
  2. **PI control functions** : PI control functions for each motor were present in this library function. Different function were made since all of the motors were not required to be move in every state.Therefore these functions helped to control each motor to call independently.The function used to take parameter of desired position of the motor and according to current position it used supply voltage(pwm) to motor.
  3. **Variables :** All the Global Variables and macros were declared in this .h file.

4. **Initialisation** : This .h file consists of Definition of Initalisation function which initialises certain variables, setting configuration of all required pins etc.This function is called in setup.

   Only that header files were included which were required for total and final Locomotion of bot.

   Following this structure helped us :
   - To keep the code clean and readable.
   - To keep the things independent and work on particular modules or pattern without affecting other modules or previous work.
   - Flow of code used to be maintained and addition and deletion of algorithm made simple.

FLOW OF THEME COMPLETION
(Images are attached below**)**



1. Straight Walking

- diagonal pattern, walk for some cycles say 12

2. TURN LEFT BY 45° [To allign bot to sanddune]

- After 12^{str} cycles left turn pattern will start.
- Yaw feedback from MPU6050 after each cycle.
- If Yaw is equal to (45 ± Thresh) then go to next task.

3. MAP SANDDUNE

- Walk straight with distance mapped from sanddune
- Take feedback of ULTRASONIC sensor.
- If distance < Thresh, Go to next task.

4. INCREASE CHASSIS HEIGHT AND SWITCH TO SANDDUNE PATTERN

- Use Type 2 diagonal pattern as mentioned before.
- call ~~set~~ sanddune states and functions.

5. LEFT TURN BY 15° (∵ till Yaw = 60°)

- once sanddune crossed take left turn by more 15°.

⇓

6. STRAIGHT WALKING FOR 4-5 cycles

- go straight so that ~~it~~ bot reaches near less slack part of rope. -

⇓

7. SHARP RIGHT TURN BY 60°

- ~~A~~ Take turn to allign bot with tussock. -

⇓

8. After reaching the foot of the mountain, and considering 50 points are scored by throwing Shagai, the bot is started by waving in front of the proximity sensor.

⇓

9. The bot takes certain cycles and starts to check the pitch. If it reaches flat mountain surface, it walks 1-2 steps more and stops. Then goes to maximum height and also actuates Uukhai mechanism to raise Gerege & COMPLETE the THEME

ALTERNATIVE IDEAS
1.ultra plus mpu before sanddune
2.mpu plus LS on sanddune
for Problem because of HC05
3.Proxy for Sanddune

OTHER PROBLEMS AND SOLUTIONS

1. **Unwanted drift in straight walking**
   The bot used to drift towards the right in straight walking pattern inspite of equal displacement of all racks. In order to overcome this problem the rack displacements were manipulated to obtain straight walk.
2. **Toppling**
   Sometimes unwanted toppling was observed during turning. The observed solution to this problem was using delay after the leg touches the ground to stabalise the bot in that direction.